

WHAT IS CLAIMED IS:

- 1 1. A computer implemented method for adjusting a priority
2 of an execution thread, said method comprising:
3 indicating that the execution thread needs a higher
4 priority by updating a user mode accessible data
5 area, the indicating performed without increasing
6 a priority corresponding to the execution thread;
7 detecting a preemption event;
8 reading the user mode accessible data area in response
9 to the detected preemption event; and
10 shifting the execution thread's priority based upon
11 the user mode accessible data.
- 1 2. The method of claim 1 further comprising:
2 updating a priority offset amount prior to the
3 execution thread entering a critical section,
4 whereupon the priority offset amount is included
5 in the user mode accessible data; and
6 whereupon the shifting of the execution thread's
7 priority is based upon the priority offset
8 amount.
- 1 3. The method of claim 2 further comprising:
2 setting a critical section flag within the user mode
3 accessible data indicating that the execution
4 thread is in the critical section; and
5 setting a priority applied flag within the user mode
6 accessible data indicating that the execution
7 thread's priority has not been shifted.

1 4. The method of claim 3 wherein the detecting, reading,
2 and shifting further comprise:

3 determining, at a kernel process, that the critical
4 section flag has been set and that the priority
5 applied flag indicates that the execution
6 thread's priority has not been shifted;
7 performing the shifting in response to the
8 determination; and
9 setting the priority applied flag indicating that the
10 thread's priority has been shifted.

1 5. The method of claim 4 further comprising:
2 setting the critical section flag within the user mode
3 accessible data indicating that the execution
4 thread is no longer in the critical section;
5 receiving a second preemption event at the kernel
6 process;
7 determining, at the kernel process, that the critical
8 section flag is no longer set and that the
9 priority applied flag indicates that the
10 execution thread's priority has been shifted;
11 re-shifting the execution thread's priority by the
12 priority offset amount in response to the
13 determination that the critical section flag is
14 no longer set and that the priority flag
15 indicates that the execution thread's priority
16 has been shifted; and
17 resetting the priority applied flag indicating that
18 the execution thread's priority is no longer
19 shifted.

1 6. The method of claim 1 further comprising:

2 comparing the execution thread's shifted priority with
3 a priority corresponding to a waiting thread;
4 preempting the execution thread if the waiting
5 thread's priority is superior to the execution
6 threads shifted priority; and
7 continuing execution of the execution thread if the
8 execution thread's shifted priority is superior
9 to the waiting thread's priority.

1 7. The method of claim 1 wherein the indicating is
2 performed in response to the execution thread entering a
3 critical code section that utilizes a shared system
4 resource.

1 8. An information handling system comprising:
2 one or more processors;
3 a memory accessible by the processors;
4 a nonvolatile storage device accessible by the
5 processors;
6 a preemptive multitasking operating system controlling
7 access by execution threads to the processors;
8 and
9 a delayed priority tool for a delayed priority boost
10 to the execution threads, the delayed priority
11 tool comprising:
12 means for indicating that the execution thread
13 needs a higher priority by updating a user
14 mode accessible data area, the indicating
15 performed without increasing a priority
16 corresponding to the execution thread;
17 means for detecting a preemption event;

18 means for reading the user mode accessible data
19 area in response to the detected preemption
20 event; and
21 means for shifting the execution thread's
22 priority based upon the user mode accessible
23 data.

1 9. The information handling system of claim 8 further
2 comprising:

3 means for updating a priority offset amount prior to
4 the execution thread entering a critical section,
5 whereupon the priority offset amount is included
6 in the user mode accessible data; and
7 whereupon the execution thread's priority shift is
8 based upon the priority offset amount.

1 10. The information handling system of claim 9 further
2 comprising:

3 means for setting a critical section flag within the
4 user mode accessible data indicating that the
5 execution thread is in the critical section; and
6 means for setting a priority applied flag within the
7 user mode accessible data indicating that the
8 execution thread's priority has not been shifted.

1 11. The information handling system of claim 10 wherein
2 the means for detecting, means for reading, and means for
3 shifting further comprise:

4 means for determining, at a kernel process, that the
5 critical section flag has been set and that the
6 priority applied flag indicates that the
7 execution thread's priority has not been shifted;

8 means for performing the shifting in response to the
9 determination; and
10 means for setting the priority applied flag indicating
11 that the thread's priority has been shifted.

1 12. The information handling system of claim 11 further
2 comprising:

3 means for setting the critical section flag within the
4 user mode accessible data indicating that the
5 execution thread is no longer in the critical
6 section;

7 means for receiving a second preemption event at the
8 kernel process;

9 means for determining, at the kernel process, that the
10 critical section flag is no longer set and that
11 the priority applied flag indicates that the
12 execution thread's priority has been shifted;

13 means for re-shifting the execution thread's priority
14 by the priority offset amount in response to the
15 determination that the critical section flag is
16 no longer set and that the priority flag
17 indicates that the execution thread's priority
18 has been shifted; and

19 means for resetting the priority applied flag
20 indicating that the execution thread's priority
21 is no longer shifted. ,

1 13. The information handling system of claim 8 further
2 comprising:

3 means for comparing the execution thread's shifted
4 priority with a priority corresponding to a
5 waiting thread;

6 means for preempting the execution thread if the
7 waiting thread's priority is superior to the
8 execution threads shifted priority; and
9 means for continuing execution of the execution thread
10 if the execution thread's shifted priority is
11 superior to the waiting thread's priority.

1 14. A computer program product stored in a computer
2 operable media for adjusting a priority of an execution
3 thread, said computer program product comprising:
4 means for means for indicating that the execution
5 thread needs a higher priority by updating a user
6 mode accessible data area, the indicating
7 performed without increasing a priority
8 corresponding to the execution thread;
9 means for detecting a preemption event;
10 means for reading the user mode accessible data area
11 in response to the detected preemption event; and
12 means for shifting the execution thread's priority
13 based upon the user mode accessible data.

1 15. The computer program product of claim 14 further
2 comprising:
3 means for updating a priority offset amount prior to
4 the execution thread entering a critical section,
5 whereupon the priority offset amount is included
6 in the user mode accessible data; and
7 whereupon the execution thread's priority shift is
8 based upon the priority offset amount.

1 16. The computer program product of claim 15 further
2 comprising:

3 means for setting a critical section flag within the
4 user mode accessible data indicating that the
5 execution thread is in the critical section; and
6 means for setting a priority applied flag within the
7 user mode accessible data indicating that the
8 execution thread's priority has not been shifted.

1 17. The computer program product of claim 16 wherein the
2 means for detecting, means for reading, and means for
3 shifting further comprise:
4 means for determining, at a kernel process, that the
5 critical section flag has been set and that the
6 priority applied flag indicates that the
7 execution thread's priority has not been shifted;
8 means for performing the shifting in response to the
9 determination; and
10 means for setting the priority applied flag indicating
11 that the thread's priority has been shifted.

1 18. The computer program product of claim 17 further
2 comprising:
3 means for setting the critical section flag within the
4 user mode accessible data indicating that the
5 execution thread is no longer in the critical
6 section;
7 means for receiving a second preemption event at the
8 kernel process;
9 means for determining, at the kernel process, that the
10 critical section flag is no longer set and that
11 the priority applied flag indicates that the
12 execution thread's priority has been shifted;

13 means for re-shifting the execution thread's priority
14 by the priority offset amount in response to the
15 determination that the critical section flag is
16 no longer set and that the priority flag
17 indicates that the execution thread's priority
18 has been shifted; and
19 means for resetting the priority applied flag
20 indicating that the execution thread's priority
21 is no longer shifted.

1 19. The computer program product of claim 14 further
2 comprising:
3 means for comparing the execution thread's shifted
4 priority with a priority corresponding to a
5 waiting thread;
6 means for preempting the execution thread if the
7 waiting thread's priority is superior to the
8 execution threads shifted priority; and
9 means for continuing execution of the execution thread
10 if the execution thread's shifted priority is
11 superior to the waiting thread's priority.

1 20. The computer program product of claim 14 wherein the
2 means for indicating is performed in response to the
3 execution thread entering a critical code section that
4 utilizes a shared system resource.